

REMARKS

The Specification has been amended to correct a typographical error. No amendments, additions or cancellations have been made to the claims. Thus, claims 1-15 remain pending in the captioned case. Further examination and reconsideration of the presently claimed application are respectfully requested.

Section 102 Rejection

Claims 1, 2, 4-7, and 9-15 were rejected under 35 U.S.C. § 102(b) as being anticipated by U.S. Patent No. 6,066,181 to DeMaster (hereinafter "DeMaster"). The standard for "anticipation" is one of fairly strict identity. A claim is anticipated only if each and every element as set forth in the claim is found, either expressly or inherently described, in a single prior art of reference. *Verdegaal Bros. v. Union Oil Co. of California*, 2 USPQ2d 1051, 1053 (Fed. Cir. 1987); MPEP 2131. DeMaster does not disclose all limitations of the currently pending claims, some distinctive features of which are set forth in more detail below.

DeMaster does not disclose code that elicits action from two different operating system environments. Present claims 1 and 13 each recite a first code portion and a second code portion. The first code portion elicits action from a Java virtual machine. The second code portion elicits action from a runtime operating environment. In addition to the first and second code portions, a third code portion is derived from the second code portion. The third code portion elicits action from a second operating environment, which is different from the runtime operating environment. Therefore the second and third code portions are configured for eliciting action from (at least) two different operating system environments.

Though DeMaster may disclose a first code portion that elicits action from a JVM (e.g., Java Classes 135; Fig. 1 of DeMaster), a second code portion that elicits action from a runtime operating environment (e.g., native code 124; Fig. 1 of DeMaster), DeMaster does NOT disclose a third code portion, which is derived from the second code portion to elicit action from a second operating environment, different from the runtime operating environment in which the second code portion is intended to run. In other words, DeMaster does not disclose the presently claimed second and third code portions, which are configured for eliciting action from two different operating system environments.

For example, DeMaster does not teach or suggest that native code 124 may be used to derive a third code portion, which can then be used to elicit action from an operating system environment that is different from the runtime operating environment in which native code 124 (i.e., the second code portion) is run. Instead (and as described in more detail below), DeMaster discloses the use of native code DLL 120. However, native code DLL 120 cannot be considered equivalent to the presently claimed "third object code portion", since DeMaster explicitly states that native code DLL 120 is created from (i.e., derived from) native code 124. If native code DLL 120 is then considered to be a "third object code portion", the third object code portion of DeMaster could only be used to elicit action from the same runtime operating environment in which native code 124 is intended to run. This is in direct contrast to the presently claimed "third object code portion" which, as noted in present claims 1 and 13, is used to elicit action from an entirely different operating system environment.

On pages 2-3 of the Office Action, the Examiner suggests that teaching can be found in Fig. 1 of DeMaster for the presently claimed second and third code portions. In particular, the Examiner suggests that the presently claimed second code portion is embodied within Java Native Interface Code Generator 100 of Fig. 1, and the presently claimed third code portion is embodied within Data Conversion Code Stubs 128 of Fig. 1. The Applicant respectfully disagrees, for at least the reasons set forth in more detail below.

First of all, the Java Native Interface Code Generator (JNICG) 100 shown in Fig. 1 of DeMaster cannot be considered equivalent to the presently claimed second object code portion. For example, the JNICG 100 of DeMaster fails to derive a third object code portion that contains instructions for eliciting actions from a second operating environment. As recited in the present claims, the second operating environment must be different from the runtime operating environment in which the second object code portion is intended to run. In contrast to the presently claimed case, DeMaster states:

The Java native interface code generator 100 reads and parses the user-derived native interface definition 142, in a known manner, and generates the appropriate Java and native language code required to implement the native capability in Java. Specifically, the Java native interface code generator 100 generates Java Classes 135 and data conversion code stubs 128... (DeMaster, column 4, lines 51-57).

DeMaster continues to describe how the "Java Classes 135 consist of classes of native methods... [and] are responsible for loading the native code DLL 120 at runtime." (DeMaster, column 4, lines 64-67, emphasis added). Likewise, DeMaster describes how the "data conversion code stubs 128... [are used to] convert and map the native data and functions between the native language and Java..." and how the native code DLL

120 is "created from the compilation of the native code 124 along with the data conversion code stubs 128..." (DeMaster, column 5, lines 10-15, emphasis added).

Even if JN1CG 100 (i.e., the alleged "second code portion") were considered to derive a "third code portion", the only conceivable "third code portions" taught by DeMaster (e.g., Java Classes 135, data conversion code stubs 128, or native code DLL 120) are only capable of eliciting actions from one particular operating environment (i.e., the runtime operating environment associated with the "native code" and "native methods" mentioned above). DeMaster fails to disclose so-called "third code portions" that could alternatively be used for eliciting actions from a second operating environment, which differs from the runtime operating environment associated with the "native code" or "native methods". In other words, DeMaster simply fails to disclose the presently claimed second and third code portions, which are configured for eliciting action from two different operating system environments. Accordingly, Applicant respectfully requests removal of this rejection as it pertains to independent claims 1, 13, and all claims dependent therefrom.

DeMaster does not disclose transforming a second code portion into a plurality of machine executable code modules, each of which is operable on only one of a plurality of operating systems. Present claim 6 defines not only first and second code portions, but the first code portion being indigenous to an operating system independent virtual execution function and the second code portion transformed into a plurality of machine executable code modules. Each of the machine executable modules is operable on only one unique operating system among a plurality of operating systems.

For at least the reasons set forth above, DeMaster does not disclose a second code portion that (i) elicits action from a runtime operating environment, and (ii) derives a third code portion, which elicits action from a second operating environment, different from the runtime operating environment. As such, DeMaster fails to disclose a second code portion that can be transformed into a third code portion (i.e., a machine executable code module), where the third code portion is operable on a different operating system environment than the second code portion. Consequently, DeMaster cannot be relied upon to teach or suggest a second code portion that can be transformed into a plurality of machine executable code modules, each of which is operable on only one (e.g., a different one) of a plurality of operating systems. Accordingly, Applicants respectfully request removal of this rejection as it pertains to independent claim 6 and all claims dependent therefrom.

Section 103 Rejection

Claims 3 and 8 were rejected under 35 U.S.C. § 103(a) as being unpatentable over DeMaster in view of "The AWT Native Interface," by Sun Microsystems, Inc. (hereinafter "Sun"). To establish a case of *prima facie* obviousness of a claimed invention, all of the claim limitations must be taught or suggested by the prior art. *In re Royka*, 490 F.2d. 981 (CCPA 1974); MPEP 2143.03. Obviousness can only be established by combining or modifying the teachings of the prior art to produce the claimed invention where there is some teaching, suggestion, or motivation to do so. *In re Fine*, 837 F.2d. 1071 (Fed. Cir. 1988); MPEP 2143.01. The cited art does not teach or suggest each and every limitation of the independent claims, some distinctive features of the current claims are set forth in more detail below.

DeMaster and Sun, either separate or in combination, do not teach or suggest code that elicits action from two different operating system environments. Present claims 3 and 8 are dependent from independent claims 1 and 6, respectively. Accordingly, they are patentably distinct over DeMaster for at least the reasons set forth in the § 102 arguments presented above. Though not specifically cited in the current rejection, Applicants assert that the reference to Sun cannot be combined with DeMaster to overcome the lack of teaching within DeMaster for independent claims 1 and 6.

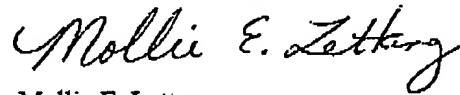
Sun discloses a Java Native Interface that "enables rendering libraries compiled to native code to draw directly to a Java Canvas drawing surface." (Sun, Abstract, page 1). More specifically, Sun discloses a "method to direct drawing operations to a native rendering library which then queries the Java VM to determine the information it needs in order to render." (Sun, How It Works, page 2). As such, the method described by Sun enables legacy software to be used in Java applications without modifying the native code of the legacy software or converting it to Java (*See, e.g.,* Sun, Introduction, page 1 and Summary, page 5). Sun, however, provides absolutely no teaching or suggestion for the presently claimed second and third code portions, which are configured for eliciting action from two different operating system environments. As such, Sun cannot be combined with DeMaster to overcome the deficiencies therein. Accordingly, Applicant respectfully request that the rejection of claims 3 and 8 be removed.

CONCLUSION

This response constitutes a complete response to all issues raised in the Office Action mailed February 12, 2004. In view of the remarks traversing the rejections, Applicant asserts that pending claims 1-15 are in condition for allowance. If the Examiner has any questions, comments or suggestions, the undersigned earnestly requests a telephone conference.

No fees are required for filing this amendment; however, the Commissioner is authorized to charge any additional fees which may be required, or credit any overpayment, to Conley Rose, P.C. Deposit Account No. 03-2769/5468-07900.

Respectfully submitted,



Mollie E. Lettang
Reg. No. 48,405
Agent for Applicant(s)

Conley Rose, P.C.
P.O. Box 684908
Austin, TX 78768-4908
(512) 476-1400
Date: May 12, 2004
JMF